# Package ‘rmetalog’

January 25, 2021

**Type** Package

**Title** The Metalog Distribution

**Version** 1.0.3

**Description** Implementation of the metalog distribution in R.
The metalog distribution is a modern, highly flexible, data-driven distribution.
Metalogs are developed by Keelin (2016) <doi:10.1287/deca.2016.0338>.
This package provides functions to build these distributions from raw data.
Resulting metalog objects are then useful for exploratory and probabilistic analysis.

**Imports** lpSolve, ggplot2

**Suggests** devtools, knitr, rmarkdown

**Depends** R (>= 3.1.0)

**BugReports** https://github.com/isaacfab/rmetalog/issues

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Isaac Faber [aut, cre] (<https://orcid.org/0000-0002-4478-9598>),
Jongbin Jung [aut]

**Maintainer** Isaac Faber <isaacfab@gmail.com>

**Repository** CRAN

**Date/Publication** 2021-01-25 18:20:02 UTC

## R topics documented:

---

`dmetalog` *Generate density values with quantiles from a metalog object. This is done through a newtons method approximation.*

---

## Description

Generate density values with quantiles from a metalog object. This is done through a newtons method approximation.

## Usage

```
dmetalog(m, q, term = 3)
```

## Arguments

| | |
|---|---|
| `m` | metalog object created from `metalog()` |
| `q` | y vector of quantiles |
| `term` | which metalog distribution to sample from |

## Value

A numeric vector of probabilities corresponding to the q quantile vector

## Examples

```
# Load example data
## Not run:
data("fishSize")

# Create a bounded metalog object
myMetalog <- metalog(fishSize$FishSize,
                     bounds=c(0, 60),
                     boundedness = 'b',
                     term_limit = 9,
                     term_lower_bound = 9)

s <- dmetalog(myMetalog,q=c(3,10,25),term = 9)

## End(Not run)
```

## fishSize — *Fish size measurements from the Pacific Northwest.*

### Description

Example data set of fish size measurements (in weight by pounds) from the Pacific Northwest, used for illustrating the flexibility of the metalog distribution. This data set is bi-modal because the fish contain two different populations, one salt and two salt runs. The two salt, fish that have gone back to the ocean twice, are typically larger.

### Usage

```
fishSize
```

### Format

A single column data frame with 3474 rows:

**FishSize** Recorded sizes of individual steelhead trout

### Source

http://www.metalogdistributions.com/

## metalog — *Fit the metalog distribution to data*

### Description

Fit the metalog distribution to data

### Usage

```
metalog(
  x,
  bounds = c(0, 1),
  boundedness = "u",
  term_limit = 13,
  term_lower_bound = 2,
  step_len = 0.01,
  probs = NA,
  fit_method = "any",
  save_data = FALSE
)
```

## Arguments

| | |
|---|---|
| `x` | vector of numeric data |
| `bounds` | numeric vector specifying lower or upper bounds, none required if the distribution is unbounded |
| `boundedness` | character string specifying unbounded, semi-bounded upper, semi-bounded lower or bounded; accepts values `u`, `su`, `sl` and `b` (default: 'u') |
| `term_limit` | integer between 3 and 30, specifying the number of metalog distributions to generate. Larger term distributions have more flexibility (default: 13) |
| `term_lower_bound` | (Optional) the smallest term to generate, used to minimize computation of unwanted terms must be less than term_limit (default is 2) |
| `step_len` | (Optional) size of steps to summarize the distribution (between 0 and 0.01) this is only used if the data vector length is greater than 100. Use this if a specific fine grid fit is required. (default is 0.01) |
| `probs` | (Optional) probability quantiles, same length as `x` |
| `fit_method` | (Optional) preferred method of fitting distribution: accepts values `OLS`, `LP` or `any` (defaults to any) |
| `save_data` | (Optional) Save the original data within the metalog object. This must be done if the distribution is to be updated with new data later. |

## Value

A `metalog` object with elements

| | |
|---|---|
| `params` | A list of the parameters used to create the metalog object |
| `dataValues` | a dataframe with the first column the raw data, second column the cumulative probabilities and the third the z vector |
| `Y` | The Y matrix values for each quantile and term |
| `A` | a dataframe of coefficients for each metalog distribution |
| `M` | a dataframe of quantiles (M) and probabilities (m) indexed for each term (i.e. M3,m3 for the third term) |
| `GridPlotCDF()` | a function that displays a grid plot of the CDF for each term |
| `VGridPlotPDF()` | a function that displays a gird plot of the PDF for each term |
| `Validation` | a vector of yes/no indicators of the valid distributions for each term |

## Examples

```
# Load example data
## Not run:
data("fishSize")

# Create a bounded metalog object
myMetalog <- metalog(fishSize$FishSize,
                     bounds=c(0, 60),
                     boundedness = 'b',
```

```
                     term_limit = 13)

## End(Not run)
```

---

plot.metalog *Plot of the metalog object*

---

## Description

Plot of the metalog object

## Usage

```
## S3 method for class 'metalog'
plot(x, ...)
```

## Arguments

| | |
|---|---|
| x | metalog object created using `metalog()` |
| ... | ignored; included for S3 generic/method consistency |

## Value

A summary plot of the CDF and PDF for each term

## Examples

```
# Load example data
## Not run:
data("fishSize")

# Create a bounded metalog object

myMetalog <- metalog(fishSize$FishSize,
                     bounds=c(0, 60),
                     boundedness = 'b',
                     term_limit = 13)

plot(myMetalog)

## End(Not run)
```

pmetalog — *Generate probabilities with quantiles from a metalog object. This is done through a newtons method approximation.*

## Description

Generate probabilities with quantiles from a metalog object. This is done through a newtons method approximation.

## Usage

```
pmetalog(m, q, term = 3)
```

## Arguments

| | |
|---|---|
| m | metalog object created from `metalog()` |
| q | vector of quantiles |
| term | which metalog distribution to sample from |

## Value

A numeric vector of probabilities corresponding to the q quantile vector

## Examples

```
# Load example data
## Not run:
data("fishSize")

# Create a bounded metalog object
myMetalog <- metalog(fishSize$FishSize,
                     bounds=c(0, 60),
                     boundedness = 'b',
                     term_limit = 9,
                     term_lower_bound = 9)

s <- pmetalog(myMetalog,q=c(3,10,25),term = 9)

## End(Not run)
```

## qmetalog — *Generate quantiles with a probability from a metalog object*

### Description

Generate quantiles with a probability from a metalog object

### Usage

```
qmetalog(m, y, term = 3)
```

### Arguments

| | |
|---|---|
| `m` | metalog object created from `metalog()` |
| `y` | vector of probabilities |
| `term` | which metalog distribution to sample from |

### Value

A numeric vector of quantiles corresponding to the y probability vector

### Examples

```
# Load example data
## Not run:
data("fishSize")

# Create a bounded metalog object
myMetalog <- metalog(fishSize$FishSize,
                     bounds=c(0, 60),
                     boundedness = 'b',
                     term_limit = 9,
                     term_lower_bound = 9)

s <- qmetalog(myMetalog,y=c(0.25,0.5,0.7),term = 9)

## End(Not run)
```

## rHDR — *Hubbard Decision Research Pseudo-Random Number Generator*

### Description

Hubbard Decision Research (HDR) Pseudo-Random Number Generator (PRNG)

## Usage

```
rHDR(
  x,
  t1 = c(2499997, 1800451, 2000371, 1796777, 2299603),
  a1 = 7450589,
  b1 = 4658,
  c1 = 7450581,
  d1 = 383,
  e1 = 99991,
  f1 = 7440893,
  t2 = c(2246527, 2399993, 2100869, 1918303, 1624729),
  a2 = 7450987,
  b2 = 7580,
  c2 = 7560584,
  d2 = 17669,
  e2 = 7440893,
  f2 = 1343
)
```

## Arguments

| | |
|---|---|
| `x` | a vector (or a ‘n * m‘ matrix) of seeds, where ‘m“ corresponds to dimensions of random numbers. Default ‘m<=5‘. See **HDR Dimensions** section below. |
| `t1, t2` | T constants (prime numbers) for 1st and 2nd term, respectively. The length of these vectors determine maximum number of dimensions for HDR PRNG. Default values are t1=c(2499997, 1800451, 2000371, 1796777, 2299603) and t2=c(2246527, 2399993, 2100869, 1918303, 1624729) |
| `a1, a2` | A constants. Default values are a1=7450589, a2=7450987 |
| `b1, b2` | B constants. Default values are b1=4658, b2=7580 |
| `c1, c2` | C constants. Default values are c1=7450581, c2=7560584 |
| `d1, d2` | D constants. Default values are d1=383, d2=17669 |
| `e1, e2` | E constants. Default values are e1=99991, e2=7440893 |
| `f1, f2` | F constants. Default values are f1=7440893, f2=1343 |

## Details

HDR PRNG is given by the formula:

$$R(x) = mod(mod(mod(10^1 5-11, mod(x*T1, A1)*B1+C1)*D1, E1)*F1+mod(mod(10^1 5-11, mod(x*T2, A2)*B2+$$

Further details on each of the dimensions

| Term | Dimension | Description | | —- | ——— | ———– | | 1 | Trial ID | This represents a unique identifier for a given scenario in a simulation. This 8 decimal digit identifier allows for up to 100 million unique trials for each variable in a modell | 2 | Variable ID | This is a unique identifier for a variable. It would be an 8-digit variable ID allowing for up to 100 million unique variables.

For example, if “Monthly Demand for Product X” and “average time spent in activity Y” were variables in a model, they would each be given unique variable IDs. Organizations may structure their Variables IDs so that related variables are in groups. For example, perhaps all marketing and sales related variables have “11” for the first two digits and all cybersecurity related variables have “73” for the first two digits, and so on. Variable IDs could be assigned on an ad hoc basis but a large organization making many models with a lot of shared variables would want to develop an internal library of assigned variable IDs similar to an accountant’s “chart of accounts.”| | 3 | Entity ID | This identifies an organization or some other category of users. A corporation or government agency may be assigned a unique 8 decimal digit Entity ID. Since this provides for 100 million potential entities, that should be enough for every business, not for profit and government agency that wants one on the planet. This is useful if there are models using random variables from many organizations do not have variables that produce the same random sequences. For example, many banks may use variables defined by the FDIC for “stress testing” to ensure banks are financially stable even during times of economic stress. The bank would want to ensure that internally defined variables with the same Variable ID are not correlated to the shared variables. The FDIC would supply the variable ID along with the Entity ID of the FDIC so that every bank using those variables produces the same sequence while avoiding duplicating the sequence of internally defined variables. A default Entity ID of 0 can be used by anyone as long as sharing variables would not be an issue.| | 4 | Time ID | This identifies a particular time unit for a given variable/trial/entity combination. This allows one scenario for a given variable to contain an entire unique time series. A 7-digit time series ID would allow for time series containing 115 days of seconds, 19 years of minutes, or 27,397 years of days. This is an optional dimension. Variables that do not represent a time series use the default Time ID of 0.| | 5 | Agent ID | This provides a fifth optional dimension for the counter-based PRNG. One possible use is as an identify for agents in agent-based modeling. If this ID is not used, the default value is 0.|

## Value

vector or pseudo-random numbers related for every one of (combination of) seeds.

## References

D. W. Hubbard, "A Multi-Dimensional, Counter-Based Pseudo Random Number Generator as a Standard for Monte Carlo Simulations," 2019 Winter Simulation Conference (WSC), National Harbor, MD, USA, 2019, pp. 3064-3073. DOI: 10.1109/WSC40007.2019.9004773

## Examples

```
rHDR(c(1:10))
rHDR(matrix(c(1:10), byrow=TRUE, nrow=5))
```

---

# rmetalog *Create random samples from an metalog distribution object*

---

## Description

The `rmetalog` package implements the metalog distribution in `R`

### Usage

```
rmetalog(m, n = 1, term = 3)
```

### Arguments

| | |
|---|---|
| m | metalog object created from `metalog()` |
| n | number of observations (default is 1) |
| term | which metalog distribution to sample from |

### Value

A numeric vector of n random samples from a selected distribution

### Examples

```
# Load example data
## Not run:
data("fishSize")

# Create a bounded metalog object
myMetalog <- metalog(fishSize$FishSize,
                     bounds=c(0, 60),
                     boundedness = 'b',
                     term_limit = 9,
                     term_lower_bound = 9)

s <- rmetalog(myMetalog, n=1000, term = 9)
hist(s)

## End(Not run)
```

---

## summary.metalog *Summary of the metalog object*

---

### Description

Summary of the metalog object

### Usage

```
## S3 method for class 'metalog'
summary(object, ...)
```

### Arguments

| | |
|---|---|
| object | metalog object created from `metalog()` |
| ... | ignored; included for S3 generic/method consistency |

### Value

A summary of the object

### Examples

```
# Load example data
## Not run:
data("fishSize")

# Create a bounded metalog object
myMetalog <- metalog(fishSize$FishSize,
                     bounds=c(0, 60),
                     boundedness = 'b',
                     term_limit = 13)

summary(myMetalog)

## End(Not run)
```

# Index